TU München, Fakultät für Informatik
Lehrstuhl III: Datenbanksysteme
Prof. Alfons Kemper, Ph.D.

Übung zur Vorlesung
Einsatz und Realisierung von Datenbanksystemen im SoSe16

Moritz Kaufmann (moritz.kaufmann@tum.de)
http://db.in.tum.de/teaching/ss16/impldb/

Blatt Nr. 10

Hinweise Die Aufgaben können auf `http://xquery.db.in.tum.de/` getestet werden. Die Daten für das Unischema können mit doc('uni2') geladen werden. Zur Lösung der Aufgaben können sie die folgenden XQuery-Funktionen verwenden:

max(NUM), count(X), tokenize(STR,SEP), sum(NUM), contains(HAY,NEEDLE)

1. max(NUMBERS) - Returns largest number from list
2. count(LIST) - Return the number of elements in the list
3. tokenize(STR,SEP) - Splits up the string at the seperator
4. sum(NUMBERS) - Returns sum of all numbers in list
5. contains(HAY,NEEDLE) - Checks if the search string (NEEDLE) is contained in the string (HAY)
6. distinct-values(LIST) - Returns the distinct values from the list

Hausaufgabe 1

Geben Sie ein Vorlesungsverzeichnis aus, welches nach dem Umfang der Vorlesungen in SWS gruppiert ist [1].

Die Ausgabe Ihrer Anfrage soll wie folgt aufgebaut sein:

```
<Vorlesungsverzeichnis>
  <Vorlesungen SWS="2">
    <Vorlesung VorlNr="V5216" Titel="Bioethik"/>
    <Vorlesung VorlNr="V5259" Titel="Der Wiener Kreis"/>
    <Vorlesung VorlNr="V5022" Titel="Glaube und Wissen"/>
    <Vorlesung VorlNr="V5049" Titel="Maeeutik"/>
  </Vorlesungen>
  <Vorlesungen SWS="3">
    <Vorlesung VorlNr="V5043" Titel="Erkenntnistheorie"/>
    <Vorlesung VorlNr="V5052" Titel="Wissenschaftstheorie"/>
  </Vorlesungen>
  <Vorlesungen SWS="4">
    <Vorlesung VorlNr="V4630" Titel="Die 3 Kritiken"/>
    <Vorlesung VorlNr="V5041" Titel="Ethik"/>
    <Vorlesung VorlNr="V5001" Titel="Grundzuege"/>
    <Vorlesung VorlNr="V4052" Titel="Logik"/>
  </Vorlesungen>
</Vorlesungsverzeichnis>
```

[1] Sie können die Aufgabe unter `http://xquery.db.in.tum.de` mit dem doc('uni2') Datensatz testen.

```
<Vorlesungsverzeichnis>
{
  for $sws in distinct-values(doc('uni2')//SWS)
  order by $sws
  return
  <Vorlesungen SWS="{$sws}">
  {
    for $vl in doc('uni2')//Vorlesung[SWS=$sws]
    order by $vl/Titel
    return <Vorlesung VorlNr="{$vl/@VorlNr}" Titel="{$vl/Titel}" />
  }
  </Vorlesungen>
}
</Vorlesungsverzeichnis>
```

Gruppenaufgabe 2

Formulieren Sie die zuvor in SQL bearbeiteten Anfragen zur Universitätsdatenbank in XQuery. Erstellen Sie insbesondere XQuery-Anfragen, um folgende Fragestellungen zu beantworten [2]:

a) Suchen Sie die Professoren, die Vorlesungen halten.

b) Finden Sie die Studenten, die alle Vorlesungen gehört haben.

c) Finden Sie die Studenten mit der größten Semesterzahl unter Verwendung von Aggregatfunktionen.

d) Berechnen Sie die Gesamtzahl der Semesterwochenstunden, die die einzelnen Professoren erbringen. Dabei sollen auch die Professoren berücksichtigt werden, die keine Vorlesungen halten.

e) Finden Sie die Studenten, die alle vierstündigen Vorlesungen gehört haben.

f) Finden Sie die Namen der Studenten, die in keiner Prüfung eine bessere Note als 3.0 hatten.

g) Berechnen Sie den Umfang des Prüfungsstoffes jedes Studenten. Es sollen der Name des Studenten und die Summe der Semesterwochenstunden der Prüfungsvorlesungen ausgegeben werden.

h) Finden Sie Studenten, deren Namen den eines Professors enthalten.

i) Ermitteln Sie den Bekanntheitsgrad der Professoren unter den Studenten, wobei wir annehmen, dass Studenten die Professoren nur durch Vorlesungen oder Prüfungen kennen lernen.

[2] Sie können die Aufgabe unter `http://xquery.db.in.tum.de` mit dem doc('uni2') Datensatz testen.

```
(: a) Suchen Sie die Professoren, die Vorlesungen halten. :)
doc('uni2')//ProfessorIn[.//Vorlesung]/Name

(: b) Finden Sie die Studenten, die alle Vorlesungen gehört haben. :)
doc('uni2')//Student[count(tokenize(hoert/@Vorlesungen," "))=count(//Vorlesung)]/Name

(: c) Finden Sie die Studenten mit der größten Semesterzahl unter
  Verwendung von Aggregatfunktionen. :)
let $maxsws:=max(data(doc('uni2')//Student/Semester))
return doc('uni2')//Student[Semester=$maxsws]

 (: d) Berechnen Sie die Gesamtzahl der Semesterwochenstunden, die die
  einzelnen Professoren erbringen. Dabei sollen auch die Professoren
  berücksichtigt werden, die keine Vorlesungen halten. :)
for $p in doc('uni2')//ProfessorIn
return <Prof>{$p/Name}<Summe>{sum(data($p//SWS))}</Summe></Prof>

(: e) Finden Sie die Studenten, die alle vierstundigen Vorlesungen
   gehört haben. :)
let $fourcount:=count(doc('uni2')//Vorlesung[SWS=4])
for $s in doc('uni2')//Student
where count(
  for $h in tokenize($s/hoert/@Vorlesungen," ")
  where doc('uni2')//Vorlesung[@VorlNr=$h and SWS=4]
  return $h
) = $fourcount
return $s/Name



(: f) Finden Sie die Namen der Studenten, die in keiner Prufung
   eine bessere Note als 3.0 hatten. :)
for $s in doc('uni')//Student
where count(
  for $p in $s//Pruefung
  where $p/@Note < 3
  return $p
) = 0
return $s

(: g) Berechnen Sie den Umfang des Prüfungsstoffes jedes Studenten.
   Es sollen der Name des Studenten und die Summe der Semesterwochenstunden
   der Prüfungsvorlesungen ausgegeben werden. :)
for $s in doc('uni')//Student
return <Student>{$s/Name}<sum>{
  sum(for $p in $s//Pruefung
      return doc('uni')//Vorlesung[@VorlNr=$p/@Vorlesung]/SWS)}
</sum></Student>
```

```
(: h) Finden Sie Studenten, deren Namen den eines Professors enthalten. :)
for $s in doc('uni')//Student
where doc('uni')//ProfessorIn[contains($s/Name,Name)]
return $s/Name


(: i) Ermitteln Sie den Bekanntheitsgrad der Professoren unter den Studenten,
   wobei wir annehmen, dass Studenten die Professoren nur durch Vorlesungen
   oder Prüfungen kennen lernen. :)
for $p in doc('uni')//ProfessorIn
return
  <Professor>
    {$p/Name}
    <Bekanntheit>
    {
      count(
        doc('uni')//Student[.//Pruefung[@Pruefer=$p/@PersNr]]/Name
        union
        ( for $v in $p//Vorlesung/@VorlNr
          return doc('uni')//Student[contains(hoert/@Vorlesungen,$v)]/Name)
      )
    }
    </Bekanntheit>
  </Professor>
```